Tecnológico de Costa Rica

Introducción a la programación

Examen II

Tiempo 1 hora con 45 minutos

Instrucciones:

- Se le presentan 4 problemas para los cuales debe desarrollar en Python los programas que los solucionen. Los 4 problemas son de algoritmos sobre listas y matrices.
- **No puede utilizar las funciones de list,** solamente puede**: in, len, append, sum, any, all, min, max**.
- **No puede utilizar map, zip.**
- Cada función debe retornar lo que se le solicita, **ninguno debe ser con prints o inputs**.
- Cada ejercicio tiene un valor de 25 puntos del examen.
- Durante los **primeros 30 minutos podrá hacer consultas** sobre el examen a los profesores o asistentes. Pasado este tiempo no se podrá consultar más.
- Utilice los nombres de la funciones **exactamente** como se le solicitan, en caso contrario no se le revisará el ejercicio.
- El uso de Internet está limitado exclusivamente para descargar y subir el examen del TEC Digital. El uso de internet, chats, modelos AI implicarán la cancelación de la prueba y obtendrá una calificación de cero, además del proceso disciplinario que el TEC indica para los casos de plagio.

Entrega:

- Debe subir un archivo py al Tec Digital, en la sección Examenes > Examen II. El nombre del archivo debe ser Ex2_Apellido_Nombre.py.
- El Tec Digital se cierra a las 9:20 a.m., no se podrán entregar exámenes pasado ese tiempo.

1. comprimir_lista (lista)

Escriba una función comprimir_lista que reciba una lista de enteros (que puede contener números repetidos consecutivamente) y devuelva una lista de listas de dos elementos, donde cada lista contiene un número de la lista original y la cantidad de veces consecutivas que aparece. (25 pts)

```
Ejemplos:

comprimirLista ( [1, 1, 1, 2, 2, 3, 4, 4, 4, 4] )
> [[1, 3], [2, 2], [3, 1], [4, 4]]

comprimirLista ( [1, 1, 1, 2, 2, 3, 4, 4, 4, 4, 2,1,1,1,1] )
> [[1, 3], [2, 2], [3, 1], [4, 4],[2,1],[1,4]]
```

2. natural (lista)

Construya una función iterativa en Python que se llame natural(lista) . Esta función recibe una lista desordenada y produce una lista de listas, donde cada sublista mantiene el orden natural de menor a mayor de la lista original.

```
Ejemplos:

natural([3,2,1])
> [[3],[2],[1]]

natural([3,4,5,1,2,3,7,8,6,8])
> [[3,4,5],[1,2,3,7,8],[6,8]]

natural ([1,2,3])
> [1,2,3]
```

3. cruces_perfectas (matriz)

Haz un algoritmo que, dada una **matriz cuadrada nxn** (compuesta solo por ceros y unos), determine la cantidad de **cruces perfectas** que existen en ella.

Una **cruz perfecta** está formada por:

- Dos líneas **perpendiculares** (una horizontal y una vertical) que se cruzan exactamente en un punto.
- Las dos líneas deben tener la **misma longitud total**.

- La cruz está compuesta **solo de 1s**.
- El punto central es el lugar donde se cruzan las dos líneas (ese punto pertenece tanto a la fila como a la columna).

La cruz puede tener cualquier tamaño (mínimo 3 celdas en cada línea), pero siempre debe ser **simétrica** en ambos ejes.

```
matriz_9x9 = [
    [0, 0, 0, 0, 1, 0, 0, 0, 0],
    [1, 0, 0, 0, 1, 0, 0, 1, 0],
    [1, 0, 0, 0, 1, 0, 0, 0, 1],
    [0, 0, 0, 0, 1, 0, 0, 0, 0],
    [1, 1, 1, 1, 1, 1, 1, 1, 1],
    [0, 0, 0, 0, 1, 0, 0, 0, 0],
    [1, 1, 0, 0, 1, 0, 1, 0, 0],
    [1, 1, 0, 1, 1, 1, 1, 1, 0],
    [1, 1, 0, 0, 1, 0, 1, 0, 0]
]
```

Salida: 3 (hay 3 cruces de unos, de diversos tamaños)

4. ceros_abajo (matriz)

cerosAbajo (matriz). Este algoritmo recibe una matriz cuadrada, retorna True si debajo de la diagonal y **solo allí,** se encuentran valores cero, en todas las posiciones. Si hay un valor diferente que cero debajo de la matriz, si hay un cero en otra celda que no sea debajo de la diagonal de la matriz, el resultado será False (25 ptos).

D	G	B	T	4	15
0	F	5	R	6	14
0	**0**	B	5	7	5
0	**0**	**0**	G	R	F
0	**0**	**0**	**0**	3	4
0	**0**	**0**	**0**	**0**	1

True Solo ceros bajo Diagonal

D	G	B	T	4	15
0	F	5	R	6	14
0	0	B	5	7	5
0	0	0	G	R	F
0	0	0	0	**0**	4
0	0	0	0	0	1

False Ceros en la diagonal

D	G	B	T	4	15
0	F	5	R	6	14
0	0	B	5	7	5
0	0	0	G	R	F
0	**5**	0	0	3	4
0	0	0	0	0	1

False Un NO cero bajo la diagonal

D	G	B	T	4	15
0	F	5	R	6	14
0	0	B	5	**0**	5
0	0	0	G	R	F
0	0	0	0	3	4
0	0	0	0	0	1

False Ceros sobre la diagonal